Signature Project: MongoDB + Python Flask Web Framework + REST API + GKE

Presented by MANICKAM RAVISEKAR ,
Master of Science in Computer Science,
19599 , Spring 2022

Guidance from Dr., Professor Henry Chang
TA : Zizhuo Huang

SAN FRANCISCO BAY UNIVERSITY
47671 WestingHouse Dr., Fremont, CA 94539

ACKNOWLEDGEMENT

One of our Master's Degree Project for

Signature Project: MongoDB + Python Flask Web Framework + REST API + GKE HTTP JSON using JavaScript for the time server was an Interesting, made me to learn new things, it is useful in designing and applying on Google Cloud – Kubernetes.

For deploying this project , I would like to thank Dr. Henry Chang and Zizhuo Huang.

Also, for all I would like to always pray to Almighty for giving us wisdom and power to understand things.

Conclusion

Application usage of student server and bookshelf:

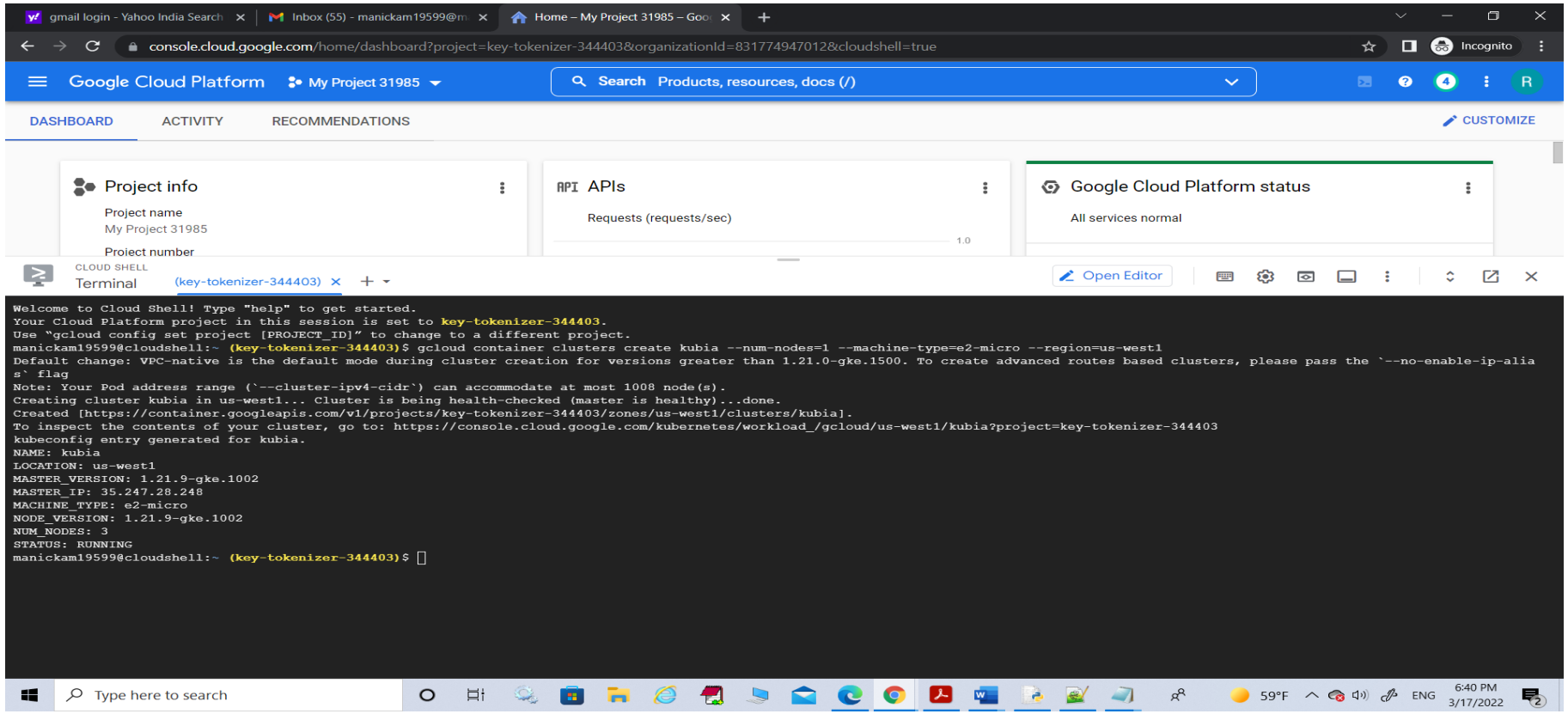
1) Student Server program is use to get students details

2) By REST API using key field of student id

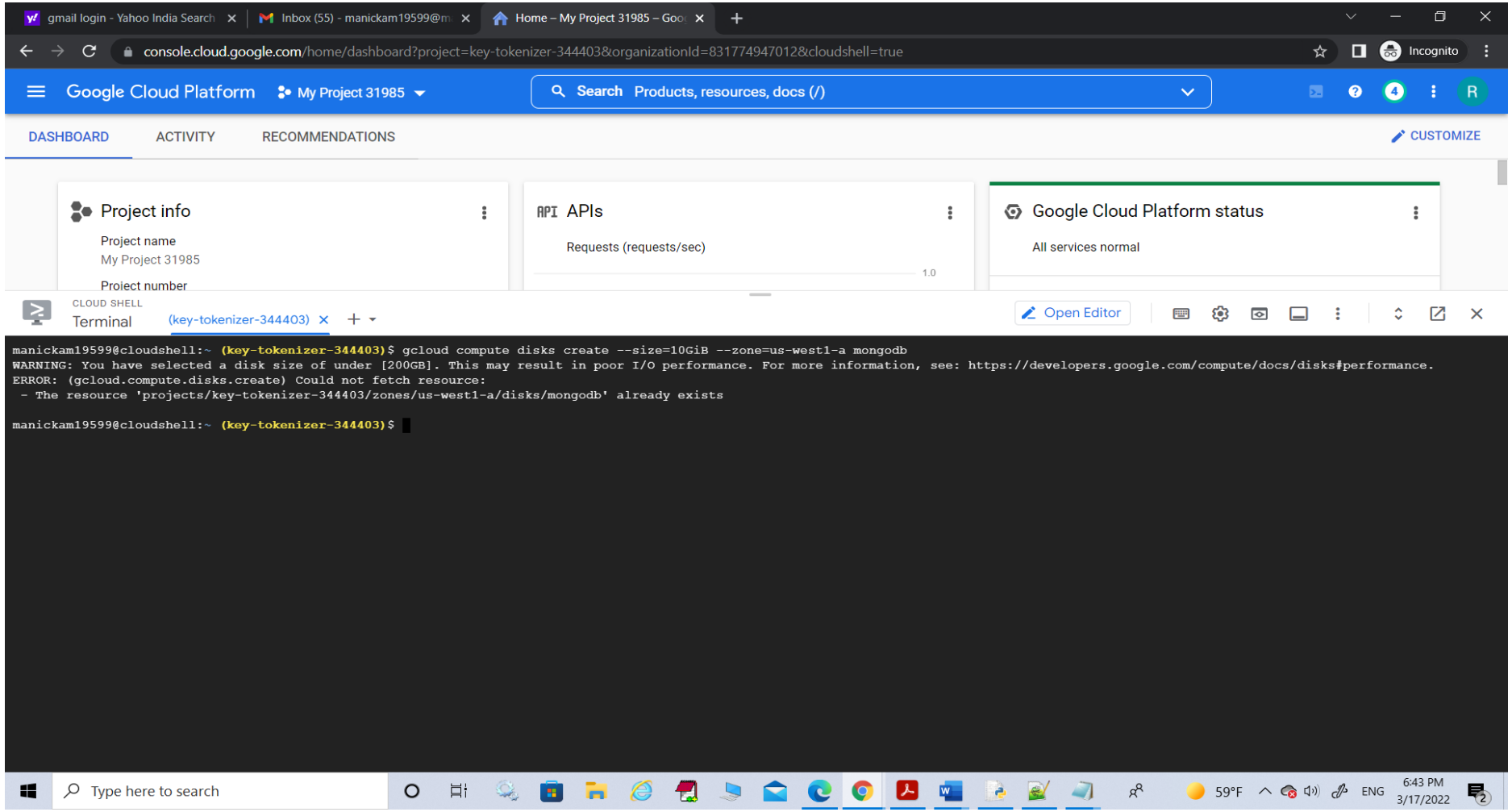
3) Book Shelf Application allows to manage books by adding new books, modifying books and deleting books .

Step1 Create MongoDB using Persistent Volume on GKE, and insert records into it :-

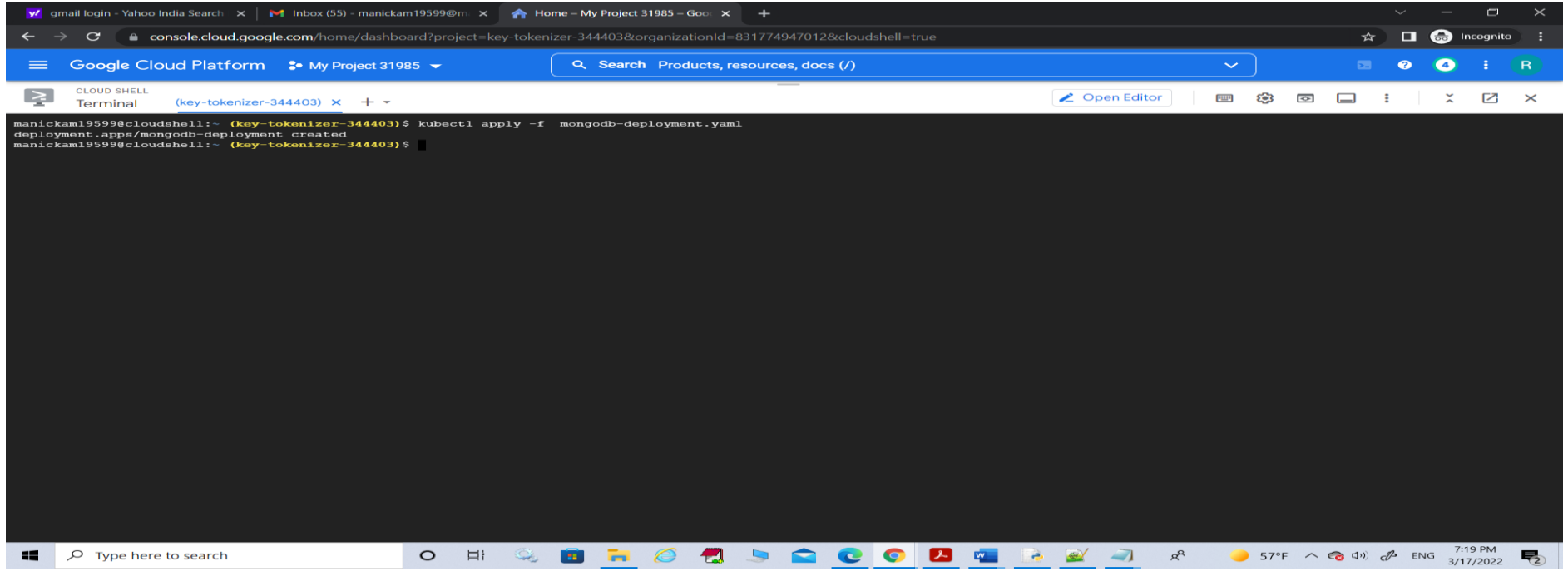
1. Create a cluster as usual on GKE
gcloud container clusters create kubia --num-nodes=1 --machine-type=e2-micro --region=us-west1



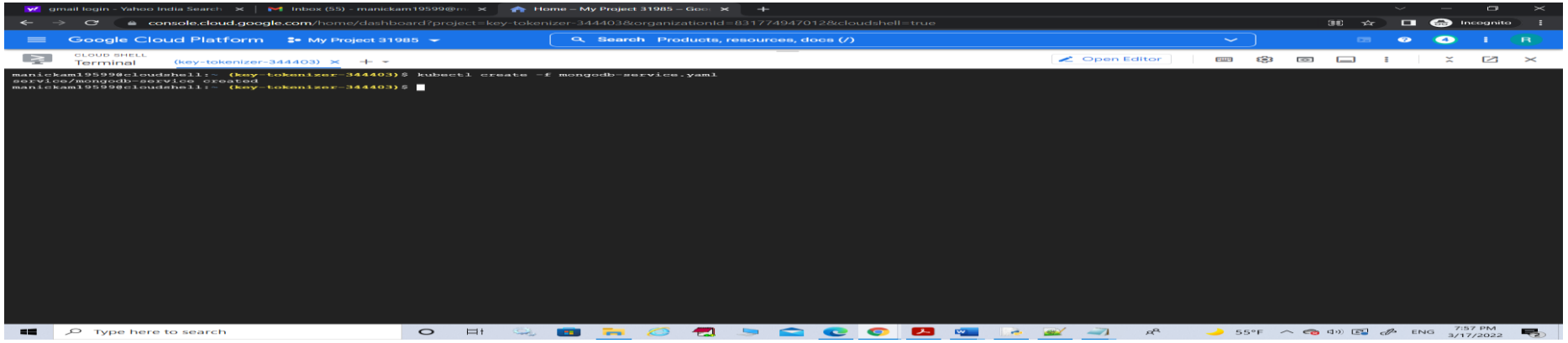
Let’s create a Persistent Volume first,
gcloud compute disks create --size=10GiB --zone=us-west1-a mongodb



kubectl apply -f mongodb-deployment.yaml: **Wait until pod starts running**

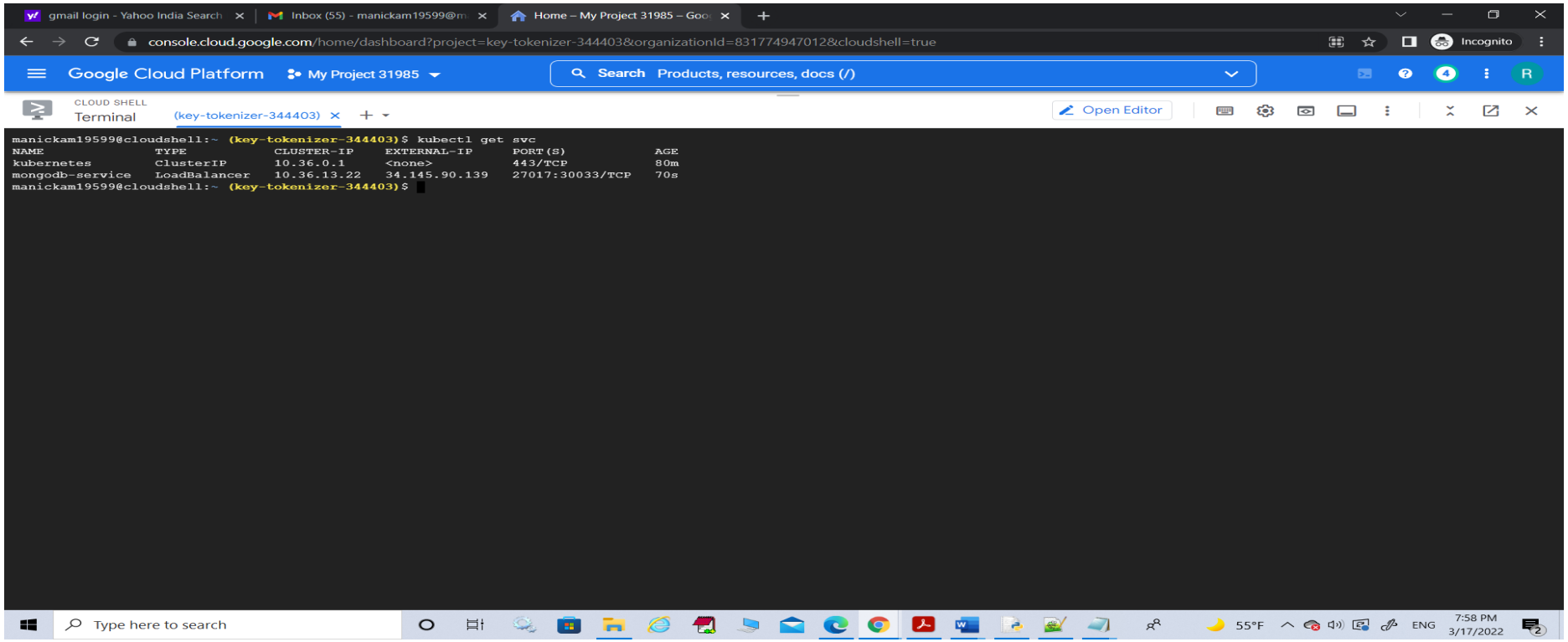


kubectl apply -f mongodb-service.yaml

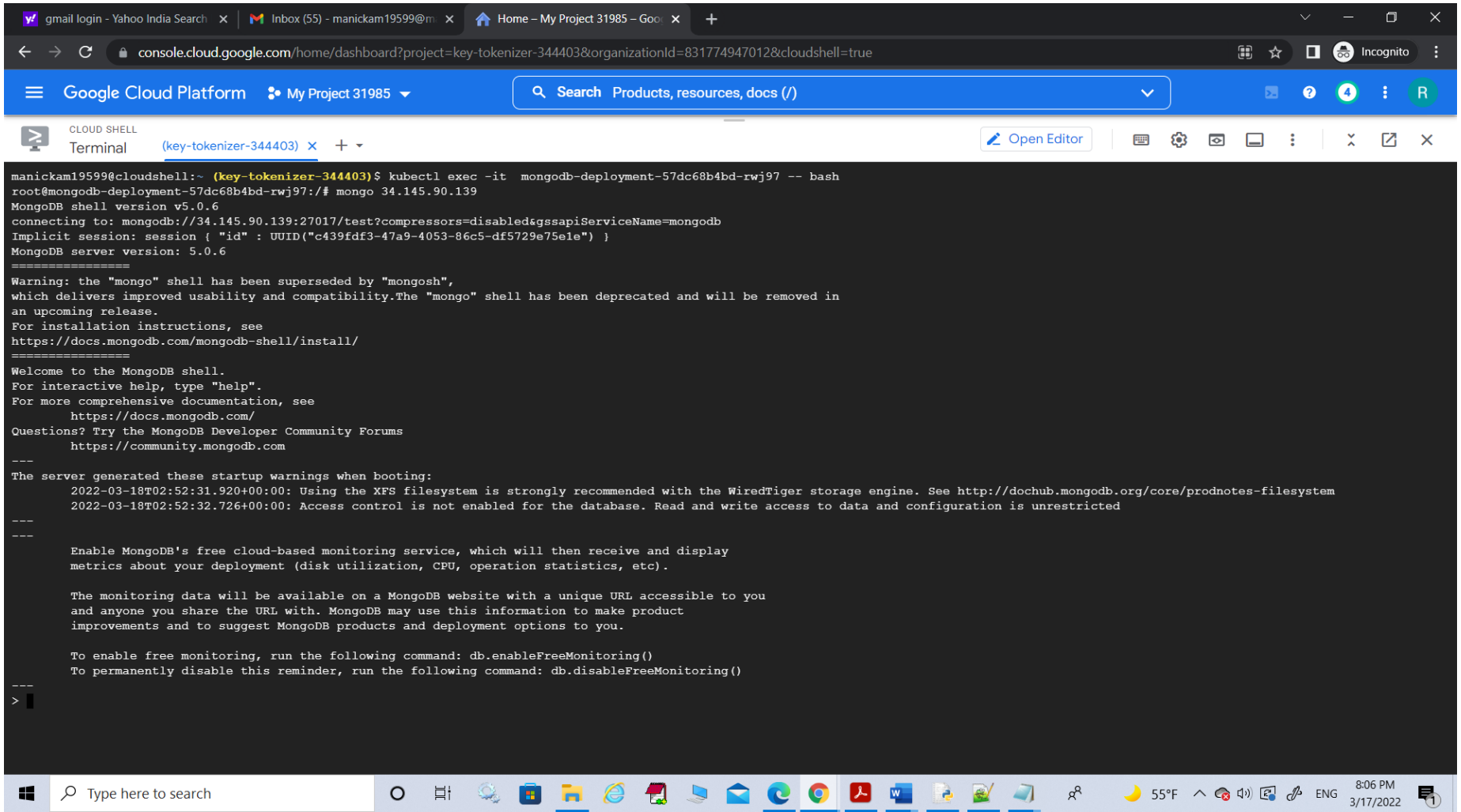


6. Wait couple of minutes, and check if the service is up

kubectl get svc Please wait until you see the external-ip is generated for mongodb-service, then you can move forward



Now try and see if mongoDB is functioning for connections using the External-IP
kubectl exec -it mongodb-deployment-replace-with-your-pod-name -- bash
Now you are inside the mongodb deployment pod
Try : mongo External-IP
You should see something like this, which means your mongoDB is up and can be accessd using the External-IP : mongo 34.145.90.139



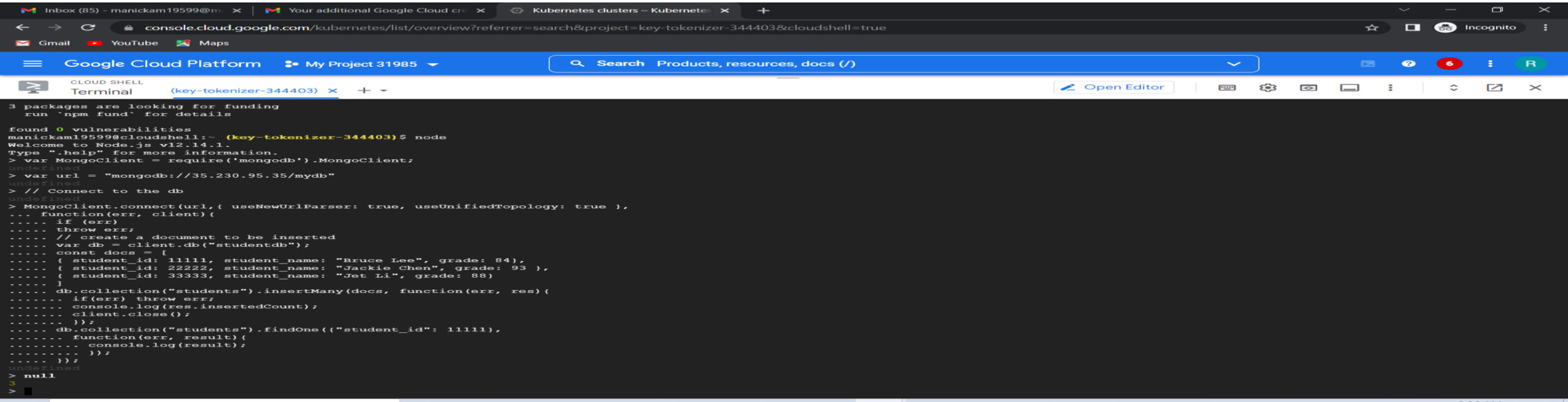
.exit

```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ kubectl exec -it  mongodb-deployment-57dc68b4bd-rwj97 -- bash
root@mongodb-deployment-57dc68b4bd-rwj97:/# mongo 34.145.90.139
MongoDB shell version v5.0.6
connecting to: mongodb://34.145.90.139:27017/test?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("c439fdf3-47a9-4053-86c5-df5729e75e1e") }
MongoDB server version: 5.0.6
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
---
The server generated these startup warnings when booting:
        2022-03-18T02:52:31.920+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
        2022-03-18T02:52:32.726+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> exit
bye
root@mongodb-deployment-57dc68b4bd-rwj97:/# exit
exit
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Enter the following line by line

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://34.82.40.170/mydb"
// Connect to the db
MongoClient.connect(url,{ useNewUrlParser: true, useUnifiedTopology: true },
function(err, client){
if (err)
throw err;
// create a document to be inserted
var db = client.db("studentdb");
const docs = [
{ student_id: 11111, student_name: "Bruce Lee", grade: 84},
{ student_id: 22222, student_name: "Jackie Chen", grade: 93 },
{ student_id: 33333, student_name: "Jet Li", grade: 88}
]
db.collection("students").insertMany(docs, function(err, res){
if(err) throw err;
console.log(res.insertedCount);
client.close();
});
db.collection("students").findOne({"student_id": 11111},
function(err, result){
console.log(result);
});
});
```

We need to insert some records into the mongoDB for later use node use : node and do the entry as given beside and check you can see 3 (records created)



Modify our studentServer to get records from MongoDB and deploy to GKE :- studentServer.js page 1 / 2

```
var http = require('http');
var url = require('url');
var mongodb = require('mongodb');
const {
  MONGO_URL,
  MONGO_DATABASE
} = process.env;
var MongoClient = mongodb.MongoClient;
var uri = `mongodb://${MONGO_URL}/${MONGO_DATABASE}`;
// Connect to the db
console.log(uri);
var server = http.createServer(function(req, res) {
  var result;
  // req.url = /api/score?student_id=11111
  var parsedUrl = url.parse(req.url, true);
  var student_id = parseInt(parsedUrl.query.student_id);
```

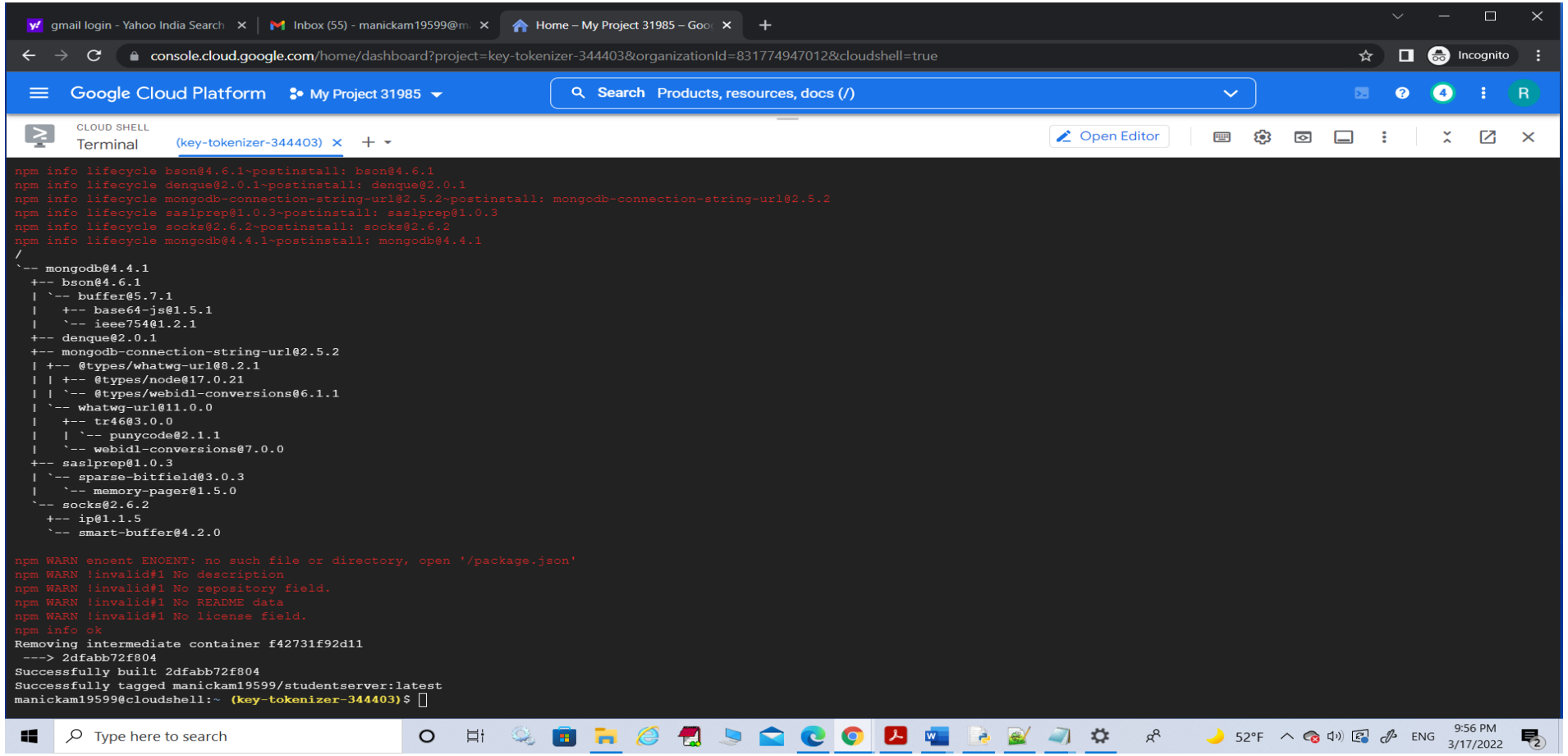
Modify our studentServer to get records from MongoDB and deploy to GKE :- studentServer.js page 2 / 2

```
// match req.url with the string /api/score
  if (/^\/api\/score/.test(req.url)) {
    // e.g., of student_id 1111
    MongoClient.connect(uri, {
      useNewUrlParser: true,
      useUnifiedTopology: true
    }, function(err, client) {
      if (err)
        throw err;
      var db = client.db("studentdb");
      db.collection("students").findOne({
          "student_id": student_id
        },
        (err, student) => {
          if (err)
            throw new Error(err.message, null);
          if (student) {
            res.writeHead(200, {
              'Content-Type': 'application/json'
            })
            res.end(JSON.stringify(student) + '\n')
          } else {
            res.writeHead(404);
            res.end("Student Not Found \n");
          }
        });     });   } else {
    res.writeHead(404);
    res.end("Wrong url, please try again\n");
  }
});
server.listen(8080);
```

Create Dockerfile

```
FROM node:7
ADD studentServer.js
/studentServer.js
ENTRYPOINT ["node",
"studentServer.js"]
RUN npm install mongodb
```

docker build -t < yourdockerhubID >/studentserver ensure no errors ignore versions warning



Push the docker image
docker push yourdockerhubID/studentserver

```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ docker push manickam19599/studentserver
Using default tag: latest
The push refers to repository [docker.io/manickam19599/studentserver]
a5d598fdc055: Pushed
38544a4882ec: Pushed
ab90d83fa34a: Mounted from library/node
8ee318e54723: Mounted from library/node
e6695624484e: Mounted from library/node
da59b99bbd3b: Mounted from library/node
5616a6292c16: Mounted from library/node
f3ed6cb59ab0: Mounted from library/node
654f45ecb7e3: Mounted from library/node
2c40c66f7667: Mounted from library/node
latest: digest: sha256:7c8eaf1c8b6ca46e6ba9d033b6c858a9a82b681952d317e1d553cb898f8a7b34 size: 2424
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Create a python Flask bookshelf REST API and deploy on GKE :- bookshelf.py - page 1 / 4

```
from flask import Flask, request, jsonify
from flask_pymongo import PyMongo
from flask import request
from bson.objectid import ObjectId
import socket
import os
app = Flask(__name__)
app.config["MONGO_URI"] = "mongodb://"+os.getenv("MONGO_URL")+"/"+os.getenv("MONGO_DATABASE")
app.config['JSONIFY_PRETTYPRINT_REGULAR'] = True
mongo = PyMongo(app)
db = mongo.db
@app.route("/")
def index():
   hostname = socket.gethostname()
   return jsonify(
   message="Welcome to bookshelf app! I am running inside {} pod!".format(hostname)
    )
@app.route("/books")
def get_all_tasks():
   books = db.bookshelf.find()
   data = []
   for book in books:
      data.append({
         "id": str(book["_id"]),
         "Book Name": book["book_name"],
         "Book Author": book["book_author"],
         "ISBN" : book["ISBN"]
      })
   return jsonify(
      data
)
```

Create a python Flask bookshelf REST API and deploy on GKE :- page bookshelf.py - 2 / 4

```
@app.route("/books")
def get_all_tasks():
    books = db.bookshelf.find()
    data = []
    for book in books:
        data.append({
            "id": str(book["_id"]),
            "Book Name": book["book_name"],
            "Book Author": book["book_author"],
            "ISBN" : book["ISBN"]
        })
    return jsonify(
        data
)

@app.route("/book", methods=["POST"])
def add_book():
    book = request.get_json(force=True)
    db.bookshelf.insert_one({
        "book_name": book["book_name"],
        "book_author": book["book_author"],
        "ISBN": book["isbn"]
        })
    return jsonify(
        message="Task saved successfully!"
)
```

Create a python Flask bookshelf REST API and deploy on GKE :-


```
@app.route("/book/<id>", methods=["PUT"])
def update_book(id):
   data = request.get_json(force=True)
   print(data)
   response = db.bookshelf.update_many ({"_id": ObjectId(id)},
            {"$set": {"book_name": data['book_name'], "book_author": data["book_author"],"ISBN":
data["isbn"]}
         }
      )
   if response.matched_count:
      message = "Task updated successfully!"
   else:
      message = "No book found!"
   return jsonify(
      message=message
   )

@app.route("/book/<id>", methods=["DELETE"])
def delete_task(id):
   response = db.bookshelf.delete_one({"_id": ObjectId(id)})
   if response.deleted_count:
      message = "Task deleted successfully!"
   else:
      message = "No book found!"
   return jsonify(
      message=message
   )
```

```
@app.route("/tasks/delete",
methods=["POST"])
def delete_all_tasks():
    db.bookshelf.remove()
    return jsonify(
        message="All Books deleted!"
    )

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

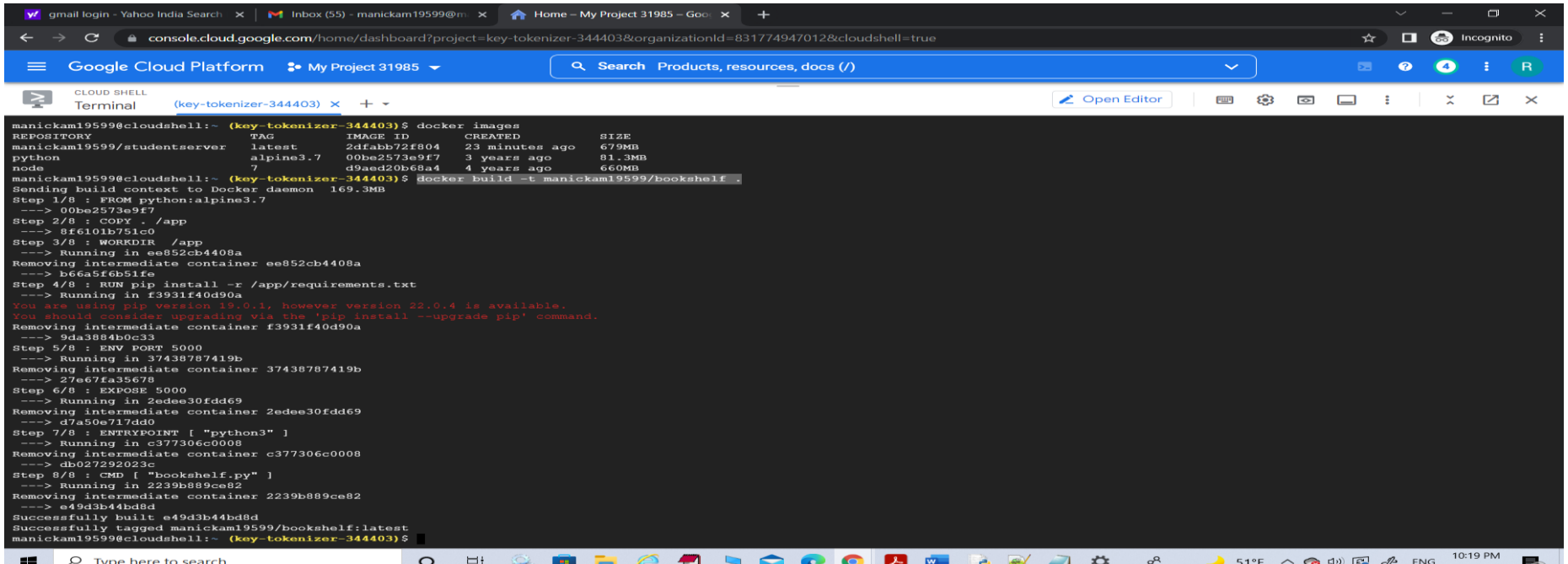
Create a Dockerfile

```
FROM python:alpine3.7
COPY . /app
WORKDIR /app
RUN pip install -r requirements.txt
ENV PORT 5000
EXPOSE 5000
ENTRYPOINT [ "python3" ]
CMD [ "bookshelf.py" ]
```

Contents of file requirements.txt

```
Flask
Flask-PyMongo
```

Build the bookshelf app into a docker image docker build -t < yourdockerhubID >/bookshelf . Make sure this step build successfully



.docker push manickam19599/bookshelf

```
Step 3/8 : WORKDIR  /app
 ---> Running in ee852cb4408a
Removing intermediate container ee852cb4408a
 ---> b66a5f6b51fe
Step 4/8 : RUN pip install -r /app/requirements.txt
 ---> Running in f3931f40d90a
You are using pip version 19.0.1, however version 22.0.4 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Removing intermediate container f3931f40d90a
 ---> 9da3884b0c33
Step 5/8 : ENV PORT 5000
 ---> Running in 37438787419b
Removing intermediate container 37438787419b
 ---> 27e67fa35678
Step 6/8 : EXPOSE 5000
 ---> Running in 2edee30fdd69
Removing intermediate container 2edee30fdd69
 ---> d7a50e717dd0
Step 7/8 : ENTRYPOINT [ "python3" ]
 ---> Running in c377306c0008
Removing intermediate container c377306c0008
 ---> db027292023c
Step 8/8 : CMD [ "bookshelf.py" ]
 ---> Running in 2239b889ce82
Removing intermediate container 2239b889ce82
 ---> e49d3b44bd8d
Successfully built e49d3b44bd8d
Successfully tagged manickam19599/bookshelf:latest
manickam19599@cloudshell:~ (key-tokenizer-344403)$ docker push  manickam19599/bookshelf
Using default tag: latest
The push refers to repository [docker.io/manickam19599/bookshelf]
b6630d562cd9: Pushed
9bbf94e51e20: Pushed
5fa31f02caa8: Mounted from library/python
88e61e328a3c: Mounted from library/python
9b77965e1d3f: Mounted from library/python
50f8b07e9421: Mounted from library/python
629164d914fc: Mounted from library/python
latest: digest: sha256:49ad796e9a24f5c425628ba79685a0929f356fccdcdf047bb95d735862ce0c33 size: 1791
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

studentserver-deployment.yaml

Expose 2 application using ingress with Nginx, so we can put them on the same Domain but different PATH

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web
  labels:
    app: studentserver-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
        - image: manickam19599/studentserver
          imagePullPolicy: Always
          name: web
          ports:
            - containerPort: 8080
          env:
            - name: MONGO_URL
              valueFrom:
                configMapKeyRef:
                  name: studentserver-config
                  key: MONGO_URL
            - name: MONGO_DATABASE
              valueFrom:
                configMapKeyRef:
                  name: studentserver-config
                  key: MONGO_DATABASE
```

studentserver-configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: studentserver-config
data:
  MONGO_URL: <Your EXTERNAL IP ADDRESS
of MONGO-DB Service>
  MONGO_DATABASE: mydb
```

studentserver-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: web
spec:
  type: LoadBalancer
  ports:
    # service port in cluster
   - port: 8080
     # port to contact inside container
     targetPort: 8080
  selector:
   app: web
```

bookshelf-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bookshelf-deployment
  labels:
    app: bookshelf-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: bookshelf-deployment
  template:
    metadata:
      labels:
        app: bookshelf-deployment
    spec:
      containers:
        - image: manickam19599/bookshelf
          imagePullPolicy: Always
          name: bookshelf-deployment
          ports:
            - containerPort: 5000
          env:
            - name: MONGO_URL
              valueFrom:
                configMapKeyRef:
                  name: bookshelf-config
                  key: MONGO_URL
            - name: MONGO_DATABASE
              valueFrom:
                configMapKeyRef:
                  name: bookshelf-config
                  key: MONGO_DATABASE
```

.**bookshelf-configmap.yaml**

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: bookshelf-config
data:
   # SERVICE_NAME.NAMESPACE.svc.cluster.local:SERVICE_PORT
  MONGO_URL: <EXTERNAL IP ADDRESS of MONGODB service>
  MONGO_DATABASE:  mydb
```

.bookshelf-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: bookshelf-service
spec:
  type: LoadBalancer
  ports:
     # service port in cluster
    - port: 5000
      # port to contact inside container
      targetPort: 5000
  selector:
    app: bookshelf-deployment
```

Key Deployments to check applications on kubernetes

.mini kube start

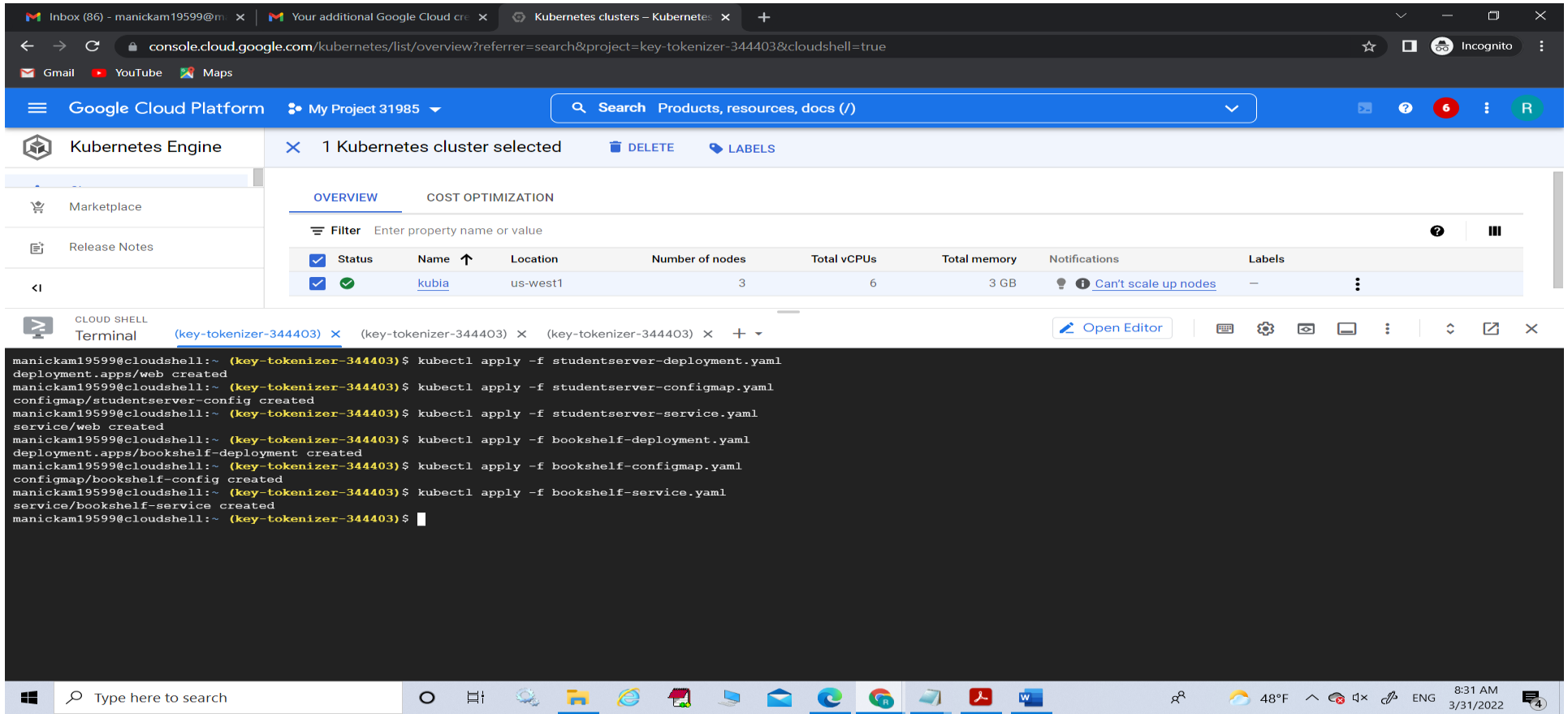
```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
mongodb-deployment-57dc68b4bd-rwj97   1/1     Running   0          3h17m
manickam19599@cloudshell:~ (key-tokenizer-344403)$ minikube start
😄 minikube v1.25.2 on Debian 11.2 (amd64)
    ▪ MINIKUBE_FORCE_SYSTEMD=true
    ▪ MINIKUBE_HOME=/google/minikube
    ▪ MINIKUBE_WANTUPDATENOTIFICATION=false
✨ Automatically selected the docker driver. Other choices: none, ssh
👍 Starting control plane node minikube in cluster minikube
🚜 Pulling base image ...
💾 Downloading Kubernetes v1.23.3 preload ...
    > preloaded-images-k8s-v17-v1...: 505.68 MiB / 505.68 MiB  100.00% 175.79 M
🔥 Creating docker container (CPUs=2, Memory=4000MB) ...
🐳 Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.cgroups-per-qos=false
    ▪ kubelet.enforce-node-allocatable=""
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎 Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟 Enabled addons: storage-provisioner, default-storageclass
🏄 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Start Ingress
minikube addons enable ingress

```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
mongodb-deployment-57dc68b4bd-rwj97   1/1     Running   0          3h17m
manickam19599@cloudshell:~ (key-tokenizer-344403)$ minikube start
😄 minikube v1.25.2 on Debian 11.2 (amd64)
    ▪ MINIKUBE_FORCE_SYSTEMD=true
    ▪ MINIKUBE_HOME=/google/minikube
    ▪ MINIKUBE_WANTUPDATENOTIFICATION=false
✨ Automatically selected the docker driver. Other choices: none, ssh
👍 Starting control plane node minikube in cluster minikube
🚜 Pulling base image ...
💾 Downloading Kubernetes v1.23.3 preload ...
    > preloaded-images-k8s-v17-v1...: 505.68 MiB / 505.68 MiB  100.00% 175.79 M
🔥 Creating docker container (CPUs=2, Memory=4000MB) ...
🐳 Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.cgroups-per-qos=false
    ▪ kubelet.enforce-node-allocatable=""
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎 Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟 Enabled addons: storage-provisioner, default-storageclass
🏄 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
manickam19599@cloudshell:~ (key-tokenizer-344403)$ minikube addons enable ingress
    ▪ Using image k8s.gcr.io/ingress-nginx/controller:v1.1.1
    ▪ Using image k8s.gcr.io/ingress-nginx/kube-webhook-certgen:v1.1.1
    ▪ Using image k8s.gcr.io/ingress-nginx/kube-webhook-certgen:v1.1.1
🔎 Verifying ingress addon...
🌟 The 'ingress' addon is enabled
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Create studentserver related pods and start service using the above yaml file
kubectl apply -f studentserver-deployment.yaml
kubectl apply -f studentserver-configmap.yaml
kubectl apply -f studentserver-service.yaml

Create bookshelf related pods and start service using the above yaml file
kubectl apply -f bookshelf-deployment.yaml
kubectl apply -f bookshelf-configmap.yaml
kubectl apply -f bookshelf-service.yaml



Check if all the pods are running correctly
kubectl get pods

console.cloud.google.com/kubernetes/list/overview?referrer=search&project=key-tokenizer-344403&cloudshell=true

Google Cloud Platform | My Project 31985 | Search Products, resources, docs (/)

Kubernetes Engine | 1 Kubernetes cluster selected | DELETE | LABELS

Marketplace

Release Notes

OVERVIEW | COST OPTIMIZATION

Filter Enter property name or value | Sort

Status	Name	Location	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
✓	kubia	us-west1	3	6	3 GB	Can't scale up nodes	–

CLOUD SHELL
Terminal (key-tokenizer-344403) (key-tokenizer-344403) (key-tokenizer-344403) | Open Editor

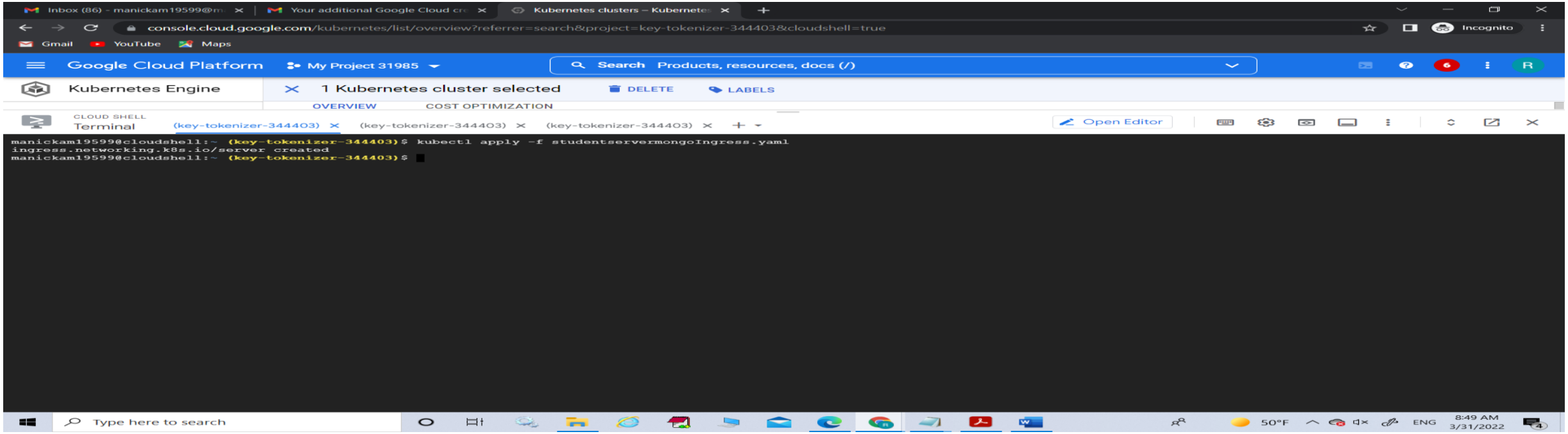
```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
bookshelf-deployment-58b84c58fc-dwsn8   1/1     Running   0          14m
mongodb-deployment-57dc68b4bd-tjn9w     1/1     Running   0          71m
web-8699589595-xcwtj                    1/1     Running   0          9m13s
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Type here to search | 48°F | ENG | 8:44 AM 3/31/2022

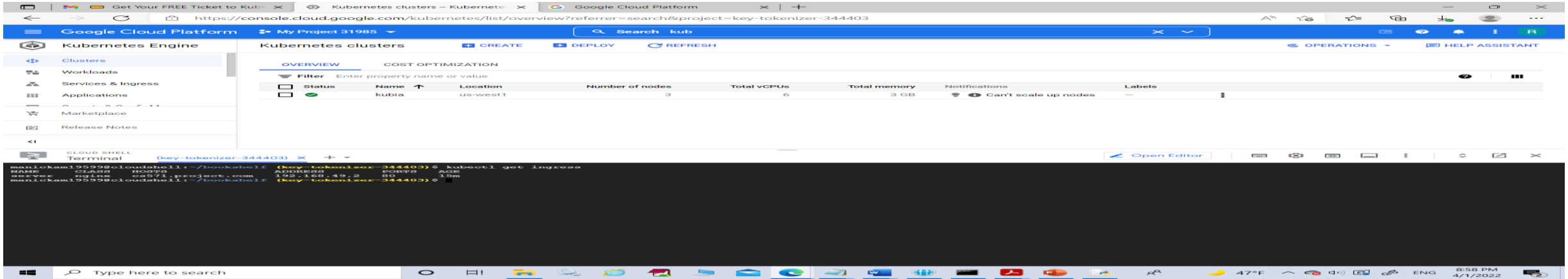
Create an ingress service yaml file called studentservermongoIngress.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: server
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$2
spec:
  rules:
    - host: cs571.project.com
      http:
        paths:
          - path : /studentserver(/|$)(.*)
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 8080
          - path: /bookshelf(/|$)(.*)
            pathType: Prefix
            backend:
              service:
                name: bookshelf-service
                port:
                  number: 5000
```

Create the ingress service using the above yaml file
kubectl apply -f ../studentservermongoIngress.yaml



Check if ingress is running
kubectl get ingress
Please wait until you see the Address, then move forward



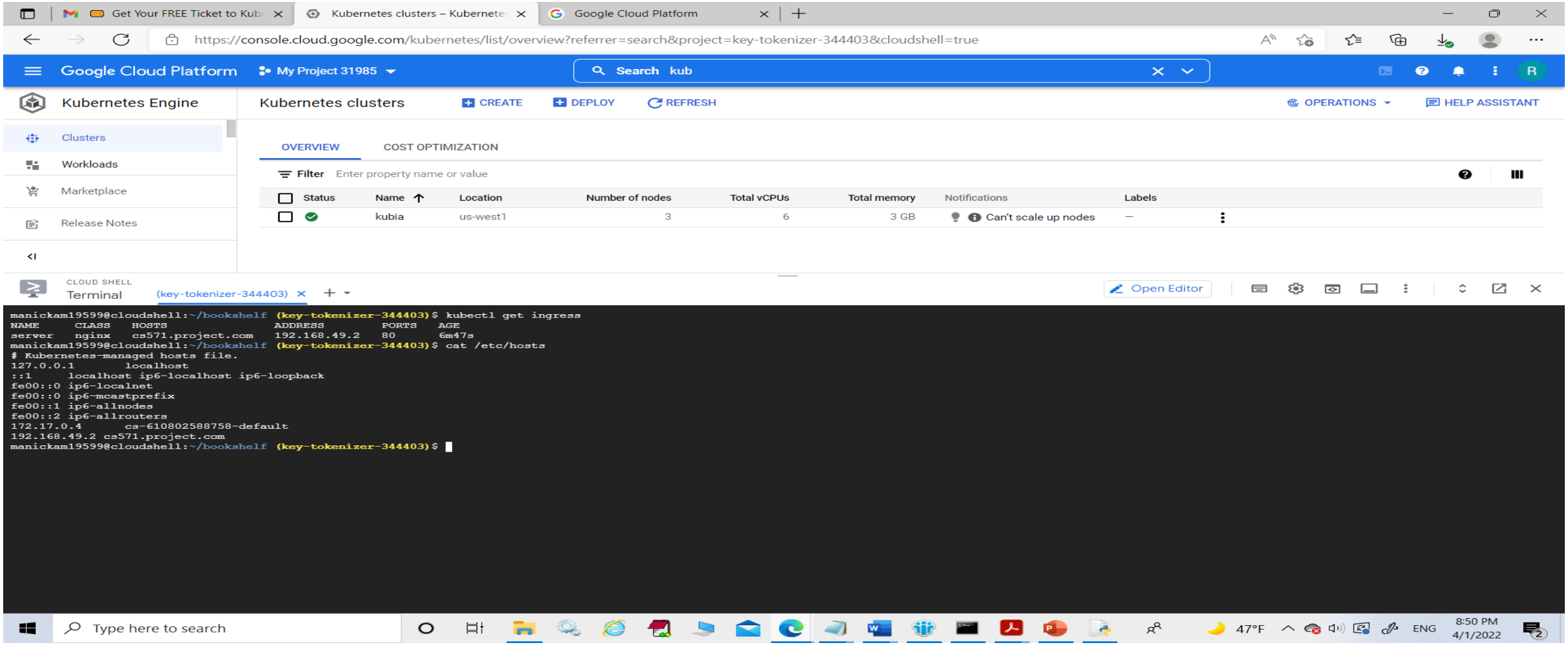
Add Addreee to /etc/hosts

vi /etc/hosts

Add the address you got from above step to the end of the file

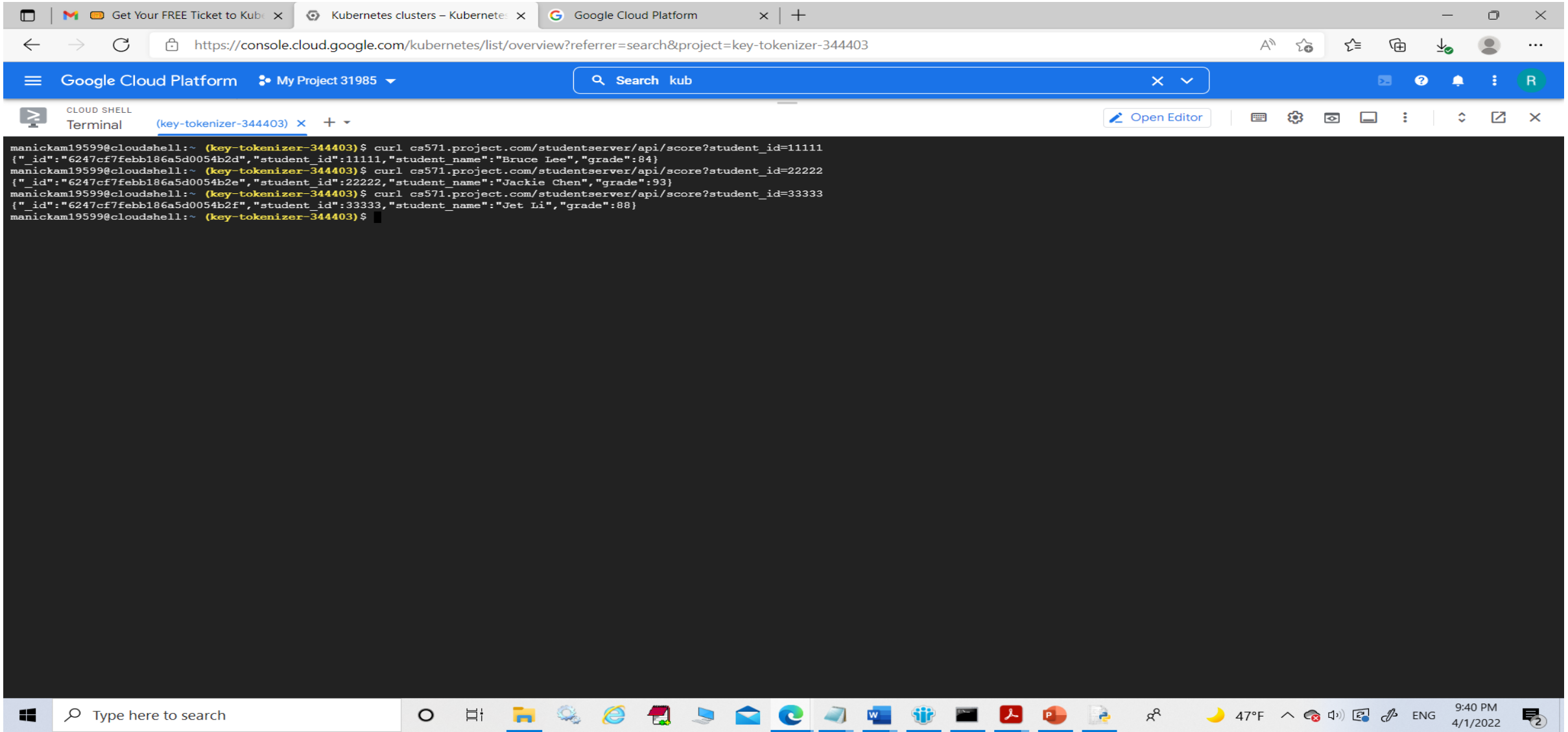
```
Your-address cs571.project.com
```

Your /etc/hosts file should look something like this after adding the line, but your address should be different from mine



If everything goes smoothly, you should be able to access your applications

curl cs571.project.com/studentserver/api/score?student_id=11111
curl cs571.project.com/studentserver/api/score?student_id=22222
curl cs571.project.com/studentserver/api/score?student_id=33333



On another path, you should be able to use the REST API with bookshelf application
I.e list all books
curl cs571.project.com/bookshelf/books

Get Your FREE Ticket to Kub | Kubernetes clusters – Kubernete | Google Cloud Platform

https://console.cloud.google.com/kubernetes/list/overview?referrer=search&project=key-tokenizer-344403

Google Cloud Platform | My Project 31985 | Search kub

Kubernetes Engine | Kubernetes clusters | CREATE | DEPLOY | REFRESH | OPERATIONS | HELP ASSISTANT

Marketplace
Release Notes

OVERVIEW | COST OPTIMIZATION

Filter Enter property name or value

Status	Name	Location	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
✓	kubia	us-west1	3	6	3 GB	Can't scale up nodes	–

CLOUD SHELL
Terminal (key-tokenizer-344403) | Open Editor

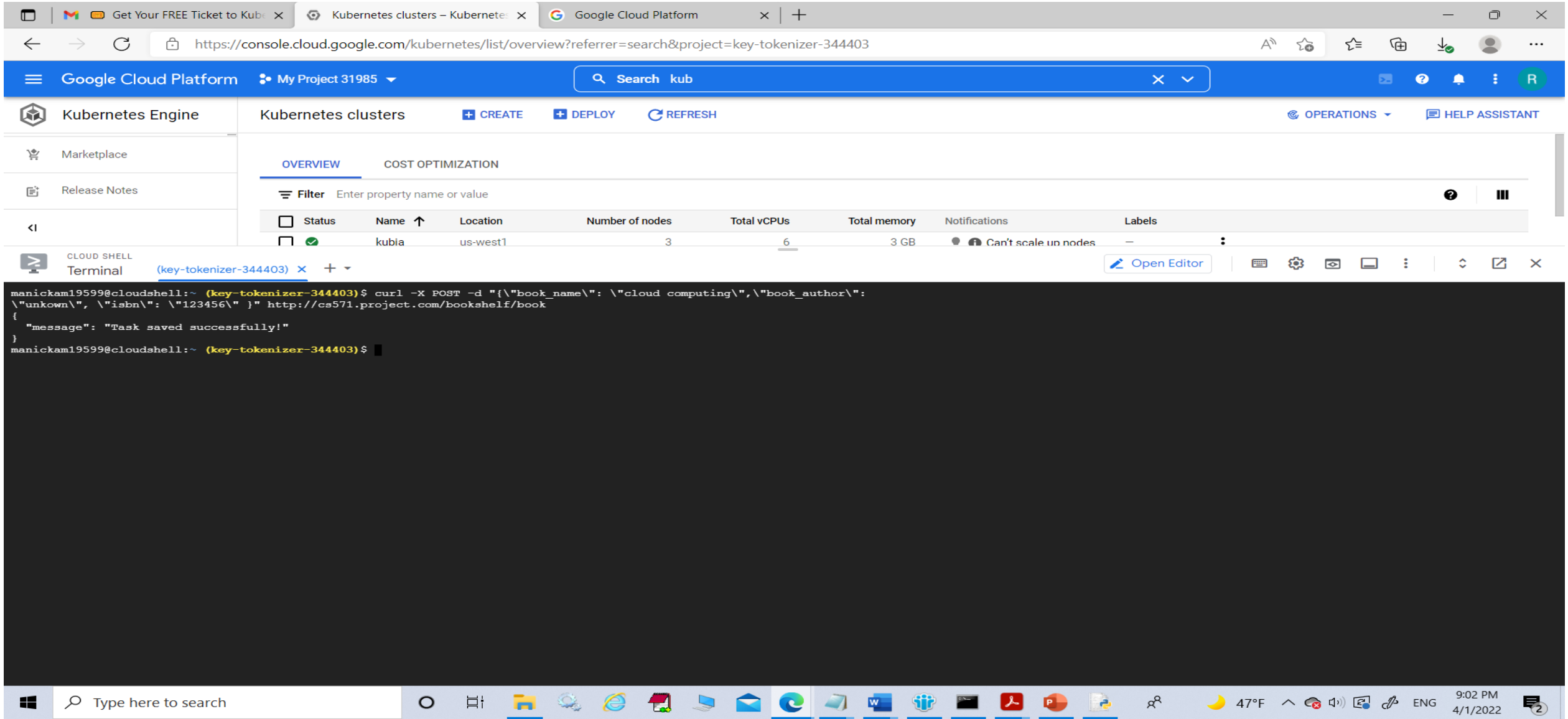
```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl cs571.project.com/bookshelf/books
[
  {
    "Book Author": "unkown",
    "Book Name": "cloud computing",
    "ISBN": "123456",
    "id": "6247caf1ebb6d7f9bd1edee4"
  }
]
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Type here to search | 47°F | ENG | 9:03 PM 4/1/2022

Add a book

curl -X POST -d "{\"book_name\": \"cloud computing\",\"book_author\": \"unkown\", \"isbn\": \"123456\" }" http://cs571.project.com/bookshelf/book

Update a



Update a book

curl -X PUT -d "{\"book_name\": \"123\",\"book_author\": \"test\", \"isbn\": \"123updated\" }" http://cs571.project.com/bookshelf/book/id

Get Your FREE Ticket to Kube | Kubernetes clusters – Kubernetes | Google Cloud Platform

https://console.cloud.google.com/kubernetes/list/overview?referrer=search&project=key-tokenizer-344403

Google Cloud Platform | My Project 31985 | Search kub

Kubernetes Engine | Kubernetes clusters | CREATE | DEPLOY | REFRESH | OPERATIONS | HELP ASSISTANT

Marketplace

Release Notes

OVERVIEW | COST OPTIMIZATION

Filter | Enter property name or value

Status	Name	Location	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
✓	kubia	us-west1	3	6	3 GB	Can't scale up nodes	–

CLOUD SHELL Terminal (key-tokenizer-344403) | Open Editor

```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl cs571.project.com/bookshelf/books
[
  {
    "Book Author": "unkown",
    "Book Name": "cloud computing",
    "ISBN": "123456",
    "id": "6247caf1ebb6d7f9bd1edee4"
  }
]
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl -X PUT -d "{\"book_name\": \"123\",\"book_author\": \"test\", \"isbn\":
\"123updated\" }" http://cs571.project.com/bookshelf/book/6247caf1ebb6d7f9bd1edee4
{
  "message": "Task updated successfully!"
}
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl cs571.project.com/bookshelf/6247caf1ebb6d7f9bd1edee4
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>404 Not Found</title>
<h1>Not Found</h1>
<p>The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.</p>
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl cs571.project.com/bookshelf/books
[
  {
    "Book Author": "test",
    "Book Name": "123",
    "ISBN": "123updated",
    "id": "6247caf1ebb6d7f9bd1edee4"
  }
]
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Type here to search | 47°F | ENG | 9:05 PM 4/1/2022

Delete a book
curl -X DELETE cs571.project.com/bookshelf/book/id

```
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl cs571.project.com/bookshelf/books
[
  {
    "Book Author": "unkown",
    "Book Name": "cloud computing",
    "ISBN": "123456",
    "id": "62468f2e0b866463e50dd707"
  }
]
manickam19599@cloudshell:~ (key-tokenizer-344403)$ curl -X DELETE cs571.project.com/bookshelf/book/62468f2e0b866463e50dd707
{
  "message": "Task deleted successfully!"
}
manickam19599@cloudshell:~ (key-tokenizer-344403)$
```

Reference

SFBU Course Material